3Т

УДК 004.6(075.8)

В.Н. Терещенко
Киевский национальный университет имени Тараса Шевченко, г. Киев, Украина
v_ter@ukr.net

Подход к решению взаимосвязанных задач геометрического моделирования

В статье рассматривается один подход к решению некоторых задач вычислительной геометрии. Этот подход позволяет путем сведения задач вычислительной геометрии к задаче построения диаграммы Вороного разработать параллельно-рекурсивный алгоритм их решения. В основе идеи алгоритма лежит схема «разделяй и властвуй».

Введение

Современное развитие компьютерных технологий позволяет ставить и решать новые сложные задачи, которые нуждаются в построении комплексных математических моделей и способов их решения. Необходимо решать на одном и том же множестве данных одновременно не одну задачу, а целую совокупность взаимосвязанных задач. Такими задачами являются задачи геометрического и графического моделирования явлений и процессов. Одной из эффективных стратегий решения отмеченных задач является определение единой общей методологии их решения в рассматриваемой области. В частности в задачах графического и геометрического моделирования сложных теплофизических процессов во многих случаях требуется точность построения модели так, как от этого принципиально зависит и точность самого решения. Так, при разработке и построении графической модели теплофизических процессов сварки двигателей самолетов чрезвычайно точной должна быть модель сварочной ванны, шва и области, предельной с ними. Дело в том, что достаточно часто в результате неоптимально выбранных режимов сварки в материале возникают микротрещины, которые в процессе эксплуатации приводят к разрушению материала, а значит, и к разрушению двигателя самолета. Для анализа таких процессов важным является определение параметров теплофизических процессов на границе расплава и материала. Чтобы получить точное решение этой задачи, необходимо решать одновременно целый комплекс теплофизических и геометрических задач, что требует значительных вычислительных ресурсов. Поэтому естественным есть вопрос разработки общих подходов, которые бы позволяли эффективно решать одновременно ряд геометрических задач на одном и том же множестве входных данных: построения политопов, триангуляции, задач близости и ряда других важных задач.

Целью данной работы является разработка единого подхода для построения визуальных моделей сложных явлений и процессов. Основой этого подхода является эффективный параллельно-рекурсивный алгоритм, который одновременно решает целую совокупность взаимосвязанных геометрических задач. Это вызвано высокими требованиями к точности разрабатываемых многозадачных моделей сложных динамических процессов. И наилучшей стратегией в этом случае будет та, которая

использует общие средства реализации задач: структуры данных, шаги выполнения процедур и представления результатов. Наиболее подходящей, на наш взгляд, может быть стратегия «разделяй и властвуй». Поскольку задачи определены на одном и том же множестве данных, то этап разбиения такого алгоритма будет общим и единым для всех задач, а на этапе слияния используется общая и единая структура данных (взвешенная сцепляемая очередь), на которой процедуры слияния выполняются быстро. Причем результаты выполнения отдельных шагов одних процедур будут использоваться другими, что и обеспечивает высокую эффективность. Важно отметить, что работа не имеет целью описать уже давно известную парадигму «разделяй и властвуй» для решения задач вычислительной геометрии. В работах [1-3] достаточно полно описаны эффективные параллельные алгоритмы решения отдельных задач вычислительной геометрии, и в том числе вышеназванная техника.

Постановка задачи. Пусть задано множество S из N точек в пространстве E^d. Разработать обобщенный эффективный рекурсивно-параллельный алгоритм одновременного решения задач вычислительной геометрии, определенных на одном и том же множестве S, нижняя оценка сложности которых – порядка $\Omega(N\log N)$ (для однопроцессорной машины).

Параллельно-рекурсивный алгоритм на основе стратегии «разделяй и властвуй»

При решении задач вычислительной геометрии основной проблемой применения схемы «разделяй и властвуй» является не линейность этапа слияния и линейная неразделимость множества точек. В рассматриваемом подходе, благодаря удачному представлению входных данных на этапе предварительной обработки и использованию параллельных вычислений на этапах разбиения и слияния, удалось построить эффективный рекурсивно-параллельный алгоритм, который снимает указанные ограничения к применению. Рассмотрим алгоритм для двумерного случая на примере задачи «все ближайшие соседи».

1. Математическая модель алгоритма

Математическая модель предлагаемого алгоритма состоит из таких основных этапов: *предварительная обработка, разбиение множества данных (рекурсивный спуск), рекурсивное слияние результатов для подмножеств (рекурсивный подъем).*

Этап 1. Предварительная обработка. Пусть заданное множество S из N точек на плоскости и $O(N)$ процессоров. На этапе предварительной обработки формируется упорядоченный массив точек $U = \{P_{ij},\ i, j = 1, N\}$. При этом индексы i, j указывают номер точки в упорядоченном списке по x и y координате соответственно. На рис. 1 показан случай для $N = 12$. Построение отсортированного списка с помощью $O(N)$ процессов можно выполнить одним из эффективных параллельных алгоритмов, детально описанных в работах [4-6], с оценками сложности порядка $O(\log^2 N)$ и $O(\log N)$. Сформированный таким способом массив подается на вход алгоритма, граф которого представлен на рис. 2. В этом графе (дереве) каждый узел обозначен целым числом k, относительно которого разбивается список точек в узлах на два равномощных, относительно медианы, списка после сравнения первых индексов элементов массива U. А каждый номер узла k определяется за один проход по дереву, если известно количество точек заданного множества.

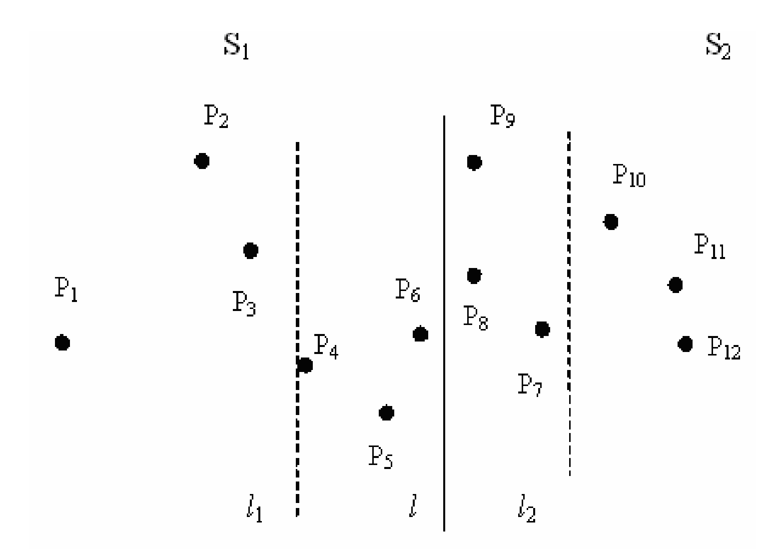


Рисунок 1 – Упорядоченный список $U = \{P_{1,3}, P_{2,12}, P_{3,9}, P_{4,2}, P_{5,1}, P_{6,5}, P_{7,8}, P_{8,11}, P_{9,6}, P_{10,10}, P_{11,7}, P_{12,4}\}$.

Этап 2. Разбиение множества точек (рекурсивный спуск). Этот этап алгоритма заключается в разбиении на каждом шаге рекурсии заданного множества точек в виде списка U на равномощные подмножества U_1, U_2, поиска медианы l и передачи U_1, U_2 на следующий шаг рекурсии. Структура данных, которая бы поддерживала этот процесс, имеет вид (рис. 2). Поиск медианы на упорядоченном по координате x индексированном массиве точек U выполняется за константное время $O(1)$: $l = (P_{kj} + P_{k+1j})/2$, где k – номер узла. Время, необходимое на рекурсивный спуск параллельного алгоритма, определяется следующей леммой.

Лемма 1. Этап рекурсивного разбиения множества U из N точек на равномощные подмножества U_1 и U_2 плоскости, поиск медианы l и передачу подмножеств на следующий уровень рекурсии с помощью $O(N)$ процессоров можно выполнить за время $O(log^2N)$.

На вход алгоритма подается множество точек в виде индексированного двумерного упорядоченного массива $U = \{P_{ij}, i,j = 1, N\}$. Для построения такой структуры данных можно воспользоваться параллельным алгоритмом сортировки со временем $O(log^2N)$, предложенным Коле [6]. Такое представление множества точек позволяет, зная количество точек N в списке U, построить дерево разбиения (рис. 2). Более детально доказательство леммы описано в работе [7].

Этап 3. Рекурсивное слияние результатов для подмножеств (рекурсивный подъем). На этом этапе, начиная с листовых узлов дерева алгоритма, полученные результаты решения задачи подаются в родительские узлы, где используются соответствующие процедуры слияния для построения общего решения задачи. Процесс завершается результатом слияния в корневом узле. В частности, эффективные процедуры слияния предложены автором в работах [7-10] для задач: построение выпуклой оболочки, диаграмма Вороного, «ближайшая пара», триангуляция. Поскольку этап разбиения общий для всех задач, то в следующем разделе предложен шаг слияния алгоритма на примере задачи «все ближайшие соседи». Особенностью предложенных процедур является применение сцепляемой очереди [11] для представления точек в узлах алгоритма, что позволяет выполнять все операции за логарифмическое время.

Вопрос использования оптимального состава процессоров (компьютеров) в данной работе не рассматривался и нуждается в последующих исследованиях. В то же время, известно из работы [12], что если взять p как количество процессоров, w – общее количество операций алгоритма, а $O(F(N))$ – сложность алгоритма для неограниченного количества процессоров, то время выполнения алгоритма можно представить в виде:

$$O(f(N))= (p - 1)O(F(N))+ w)/p. \quad (1)$$

Кроме этого, на наш взгляд, синхронная обработка данных алгоритма может быть заменена конвейерной, что расширяет возможности повышения общей эффективности алгоритма.

2. Построение процедуры слияния для задачи «все ближайшие соседи»

В работах [1], [13] после детального анализа качественных возможностей диаграммы Вороного сделан достаточно важный вывод, который можно сформулировать в виде следующего обобщенного утверждения:

Теорема 1. С помощью диаграммы Вороного для множества S из N точек на плоскости, задачи: ближайшая пара, все ближайшие соседи, евклидово минимальное остовое дерево, триангуляция, выпуклая оболочка – можно решить за оптимальное время $\theta(N\log N)$ на однопроцессорной машине.

Детальное доказательство этого результата для однопроцессорной вычислительной машины приведено в одноименной работе [13]. В основе доказательства лежит принцип сводимости. Этот вывод позволяет применить диаграмму Вороного как инструмент для решения широкого класса задач вычислительной геометрии. В данной работе этот инструмент используется при построении процедуры слияния для задачи «все ближайшие соседи».

Алгоритм построения процедуры слияния

На каждом шаге рекурсивного подъема, начиная со второго, на вход родительского узла v графа алгоритма подаются диаграммы Вороного (ДВ) от левого и правого сыновей $vor(S_L)$, $vor(S_R)$, а также ближайший сосед каждой точки подмножеств S_L, S_R. Строится диаграмма Вороного для узла v и параллельно определяются новые соседи на границах подмножеств S_L, S_R, относительно медианы l. На этапе слияния алгоритма, во время построения разделяющей цепи диаграммы Вороного, параллельно осуществляется поиск ближайших соседей среди точек множеств S_L и S_R, которые образуют текущую пару ребра цепи, $\sigma(S_L, S_R)$. При этом следующая пара, определяющая новое ребро разделяющей цепи проверяется на наличие нового ближайшего соседа. Обозначим как $NN(S)$ множество пар ближайших соседей для множества S, а $NN(S_L)$, $NN(S_R)$ – для множеств S_L и S_R соответственно. Граф алгоритма в данном случае – это двоичное дерево (рис. 2), с той лишь разницей, что каждый узел дерева загружается кроме упорядоченного массива точек медианы l диаграммами Вороного левого $vor(S_L)$ *и* правого $vor(S_R)$ сыновей, множествами соседских пар точек сыновей $NN(S_L)$ и $NN(S_R)$ соответственно.

Лемма 2. Этап рекурсивного слияния результатов определения ближайших соседей для каждой из N точек множества S на плоскости с помощью $O(N)$ процессоров можно выполнить за время $O(\log^2 N)$.

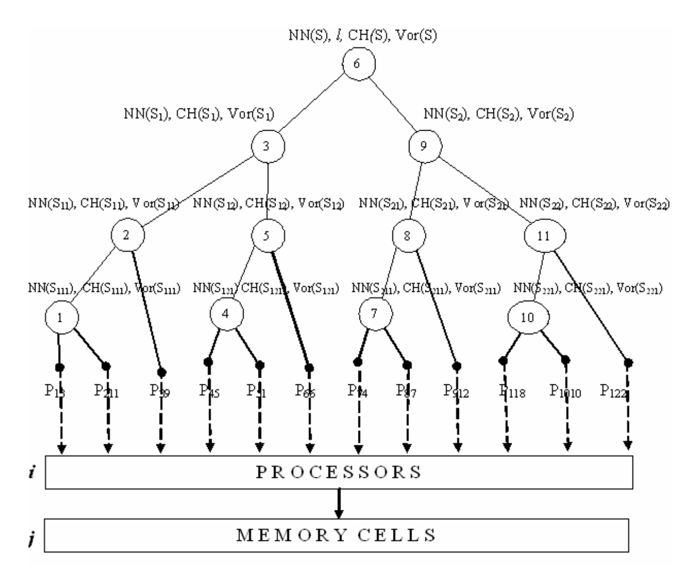


Рисунок 2 – Граф алгоритма. *NN*(*S*), *Vor*(*S*), *CH*(*S*) – процедуры слияния; *l* – медиана

Доказательство: **procedura** (СЛИЯНИЕ_ ВСЕ БЛИЖАЙШИЕ СОСЕДИ)

Для построения слияния результатов поиска ближайших соседей для узла *v*, необходимо:

1. Определить верхние и нижние опорные вершины левой и правой выпуклых оболочек множества точек сыновей S_L и S_R, а следовательно, и верхнее и нижнее опорные ребра соответственно.

2. Верхние опорные вершины (q_{BOL}, q_{BOR}) образуют первую пару точек (рис. 3), для которых необходимо выполнить следующие действия:

– взять верхнюю левую опорную точку q_{BOL} множества S_L и сравнить расстояние $\delta_1 = dist(q_{BOL}, q_{BOR})$ от нее к верхней правой опорной точке q_{BOR} множества S_R с расстоянием $\delta_{BOL} = dist(q_{BOL}, P_L)$ к ее ближайшему соседу P_L, найденного на предыдущих шагах;

– если это расстояние меньше, то ближайшим соседом для точки q_{BOL} становится точка q_{BOR} с расстоянием δ_1;

– иначе – ближайший сосед точки q_{BOL} не изменяется;

– аналогично определяется ближайший сосед точки q_{BOR}.

3. Провести входящее и исходящее ребро разделяющей цепи для опорных отрезков к пересечению с одним из ребер диаграмм $vor(S_L)$ или $vor(S_R)$.

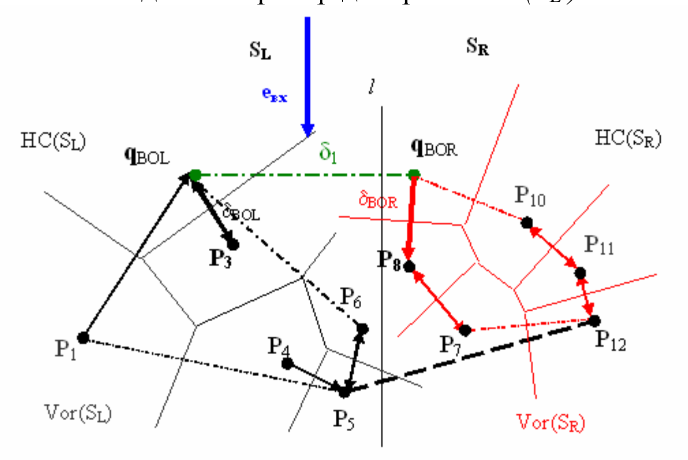


Рисунок 3 – Проверка на наличие ближайших соседей для пары точек (q_{BOL}, q_{BOR})

4. С использованием $O(N)$ процессоров проводится процесс нахождения ближайших соседей (аналогично пункту 2) для следующих пар точек (P_{iL}, P_{jR}) (где $P_{iL} \in S_L$, $P_{jR} \in S_R$, $i, j = 1, N$), которые определяются соответствующими ребрами разделяющей цепи $\sigma(S_L, S_R)$. Цепь строится для приграничных множеств точек относительно медианы l, расположенных слева и справа от нее, которые принадлежат взаимовыпуклым цепям выпуклых оболочек сыновей, и точек, определенных ребрами диаграмм $vor(S_L)$, $vor(S_R)$, которые пересекают ребра этих цепей (рис. 4).

5. Полученные после слияния таким способом множества пар ближайших соседей и соответствующие диаграммы Вороного передаются на следующий уровень рекурсии.

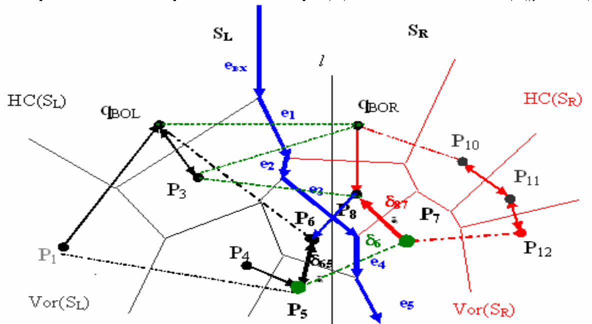


Рисунок 4 – Проверка на наличие ближайших соседей для пар точек (P_3, q_{BOR}), (P_3, P_8), (P_6, P_8), (P_6, P_7), (P_5, P_7). Для $P_8 \in S_R$ найден новый ближайший сосед $P_6 \in S_L$

6. Процесс слияния завершается, когда будет построена разделяющая цепь для корня дерева, и тем самым определено все множество пар ближайших соседей $NN(S)$ множества точек S.

Поскольку для построения монотонной разделяющей цепи диаграммы Вороного на шаге слияния с помощью $O(N)$ процессоров достаточно $O(logN)$ времени [12], то и время слияния результатов нахождения ближайших соседей будет того же порядка. Это следует из того, что для нахождения ближайшего соседа к точкам соответствующей пары, которую определяет ребро разделяющей цепи, необходимо лишь сделать два сравнения расстояний. Таким образом, можно сделать вывод.

Теорема 2. Задачу нахождения всех ближайших соседей для каждой точки множества S из N точек на плоскости можно решить с помощью $O(N)$ процессоров со временем $O(log^2N)$, с использованием $O(NlogN)$ памяти за два рекурсивных прохода параллельно-рекурсивным алгоритмом.

Реализация алгоритма

Что касается практической реализации разработанного параллельно-рекурсивного алгоритма, то одним из эффективных подходов применен подход на основе использования технологии ПАРУС (параллельные асинхронные рекурсивно управляемые системы) [14], которая позволяет достаточно просто и эффективно реализовать именно рекурсивно-параллельные алгоритмы решения сложных задач как на многопроцессорных машинах, так и в компьютерной сети. Алгоритм также успешно реализован в среде MPI.

Выводы

В работе на примере сведения задачи «все ближайшие соседи» к построению диаграммы Вороного предложен обобщенный параллельно-рекурсивный алгоритм решения ряда задач вычислительной геометрии, которые сводятся к диаграмме. Эффективность разработанного алгоритма достигается благодаря построению структу-

рированного массива точек заданного множества и удачно выбранных структур данных: дерева алгоритма и сцепленной очереди, которая описывает циклически упорядоченное множество точек и ребер многоугольников (многогранников) в каждом узле дерева алгоритма.

Этот алгоритм позволяет получить эффективность решения задач построения выпуклой оболочки, задач близости, декомпозиции порядка NC_2. Характерной чертой практической реализации данного подхода является то, что разработанный алгоритм позволяет одновременно решать как разные шаги одной процедуры задачи на многих процессорах, так и разные процедуры в одном узле алгоритма.

Литература

1. Parallel computational geometry / A. Aggarwal, B. Chazelle, L. Guibas, C. O'Dunlaing, C.C. Yap // Algorithmica. – 1988. – № 3. – P. 293-327. – New York : Springer-Verlag Inc.
2. Amato N.M. Parallel algorithms for higher dimensional conver hulls / N.M. Amato, M.T. Gudrich, E.A. Ramos // Proc. 35th Annu. IEEE Sympos. Found. Comput. Sci. – 1994. – P. 683-694
3. Atallah M.J. Parallel computational geometry // Parallel Computations: Paradigms and Applications / M.J. Atallah, D.Z. Chen ; [editore : A.Y. Zomaya]. – Boston : International Thomson Computer Press,1996.
4. An O(nlog(n)) Sorting Network / M. Ajtai, J. Komlos, E. Szemeredi // Proc. 15th ACM [«Symposium on Theory of Computing»]. – P.1-9; Combinatorica. – 1983. – № 3(1). – P. 1-19.
5. Leighton T. Tight bounds on the complexity of parallel sorting. / T. Leighton // Proc. 16th ACM Symposium on Theory of Computing. – P. 71-80.
6. Cole R. Parallel merge sort / R. Cole // Proc. 27th IEEE FOCS Symposium. – 1986. – P. 511-516.
7. Терещенко В.М. Один підхід у розробці ефективного рекурсивно-паралельного алгоритму побудови опуклої оболонки множини точок на площині. / В.М. Терещенко // Таврический вестник информатики и математики. – 2007. – № 2. – С. 24-32.
8. Терещенко В.М. Построение рекурсивно-параллельных алгоритмов решения задач вычислительной геометрии на основе стратегии «распределяй и властвуй» / В.М. Терещенко // Труды международной научной конференции «Параллельные вычислительные технологии» (ПаВТ'2008) (Санкт-Петербург, 28 января – 1 февраля 2008 г.). – Санкт-Петербург. – С. 476-482.
9. Терещенко В.М. Узагальнений підхід розв'язання деяких задач обчислювальної геометрії на основі рекурсивно-паралельної технології / В.М. Терещенко // Наукові нотатки. – 2008. – Вип. № 22 (Частина 2). – С. 344-349.
10. Построение обобщенной стратегии решения некоторых задач вычислительной геометрии / В.Н. Терещенко, В.В. Сапсай, И.С. Статкевич, А. Федоров // Труды Международной научной конференции «Параллельные вычислительные технологии» (ПаВТ'2009) (Нижний Новгород, 30 марта – 3 апреля 2009 г.). – Нижний Новгород. – С.732-743.
11. Overmars M.H. Maintenance of configurations in the plane / M.H. Overmars, J. van Leeuwen // J.Cjmput. and Syst. Sci. – 1981. – № 23. – P. 166-204
12. Воеводин В.В. Параллельные вычисления / В.В. Воеводин, Вл.В. Воеводин. – СПб. : БХВ-Петербург, 2004. – 608 с.
13. Препарата Ф. Вычислительная геометрия: Введение / Препарата Ф.М. : Мир, 1989. – 478 с.
14. Анисимов А.В. Управляющие пространства в асинхронных параллельных вычислениях / А.В. Анисимов, В.М. Глушков // Кибернетика. – 1980. – № 5. – С. 1-9.

В.Н. Терещенко

Підхід до розв'язання взаємозв'язаних задач геометричного моделювання

У роботі розглядається один підхід розв'язання деяких задач обчислювальної геометрії. Цей підхід дозволяє шляхом зведення задач близькості та опуклої оболонки до діаграми Вороного розробити паралельно-рекурсивний алгоритм їх розв'язання. В основі ідеї алгоритму лежить техніка «розподіляй та пануй». Враховуючи те, що перший етап алгоритму спільний для усіх задач, то в роботі завершальний етап алгоритму продемонстровано на прикладі задачі «усі найближчі сусіди».

V.N. Tereshchenko

The Approach to Solving the Interrelated Problems of Geometric Modelling

In the paper one approach to solution of some problems of computational geometry is considered. This approach allows us to develop a parallel-recursive algorithm for solving the problem of proximity and the problem of the convex hull, reducing them to Voronoi diagram. The algorithm bases on the «divide-and-conquer» technique. The first, «divide», stage is common for all problems so for a problem «all nearest neighbors» the procedure of merge is offered.

Статья поступила в редакцию 29.05.2009.